**Chapter 9**

Design YouTube

***Design Scope***

*YouTube Statistics*

* Total monthly active users – 2 billion
* 5 billion videos watched per day
* 73% of US adults use YouTube
* 50 million creators on YouTube
* YouTube’s ad revenue was $15.1 billion for the full year 2019, up 36% from 2018
* YouTube is responsible for 37% of all mobile internet traffic
* YouTube is available in 80 different languages

*Back of the envelope estimation*

* 5 million daily active users
* Users watch 5 videos per day
* 10% of users upload 1 video per day
* Assume the average video size = 300MB
* Daily storage space need = 5 million \* 10% \* 300 MB = 150 TB
* CDN
  + When a cloud CND serves a video, you are charged for data transferred
  + Assume 100% of traffic is served from the United States
  + 5 million \* 5 videos \* 0.3 GB \* $0.02 / GB = $150,000 per day

*A table with black text

Description automatically generated*

***High Level Design***

* Use CDN and blob storage cloud services instead of building from scratch
  + System design interview are not about building everything from scratch
  + Choosing the right technology to do a job right is more important
  + Building scalable blob storage and CDN is complex and costly
  + Even large companies like Netflix or Facebook do not build everything themselves

*Video uploading flow*

A diagram of a server

Description automatically generated

* API servers – all user requests go through API servers except video streaming
* Metadata DB – stores video metadata. It is sharded and replicated to meet performance and high availability requirements
* Metadata cache – for better performance, video metadata and user objects are cached
* Original storage – blob storage is used to store original videos.
  + BLOB – “binary large object” is a collection of binary data
* Transcoding servers – transforms video formats
  + Video transcoding = video encoding
  + The process of converting a video format to another (MPEG, HLS, etc.), which provides the best video streams possible for different devices and bandwidth capabilities
* CDN – caches videos
* Completion queue – message queue that stores info about video transcoding completion events
* Completion handler – consists of workers that pull event data from complete queue and update metadata cache and database

*Video streaming flow*

* Streaming protocol – standardized way to control data transfer for video streaming
  + MPEG-DASH
    - MPEG – Moving Picture Experts Group
    - DASH – Dynamic Adaptive Streaming over HTTP
  + Apple HLS
    - HLS – HTTP Live Streaming
  + Microsoft Smooth Streaming
  + Adobe HTTP Dynamic Streaming (HDS)
* Different streaming protocols support different video encodings and playback players

***Design Deep Dive***

*Video transcoding*

* A video must be transcoded into compatible bitrates and formats to play
  + Bitrate – the rate at which bits are processing,
  + Higher bitrate generally higher quality
* Raw video consumes large amount of storage space
* Many devices and browser only support certain types of video formats
* It is a good idea to deliver higher resolution video to users who have high network bandwidth and lower resolution video to low bandwidth
* For smooth user experience, switch video quality automatically based on network conditions is essential
* Most of encoding formats contain 2 parts
  + Container – basket containing the video file, audio, and metadata
    - Avi, mov, mp4
  + Codecs – compression and decompression algorithms aim to reduce the video size while preserving the video quality
    - H.264, VP9, and HEVC

*Directed acyclic graph (DAG) model*

* To support different video processing pipelines and maintain high parallelism
  1. Need abstraction and let client programmers define what tasks to define
     + Facebook’s streaming video engine uses DAG programming model
  2. DAG – defines tasks in stages to be executed sequentially or parallelly

A diagram of a computer program

Description automatically generated

*Video Transcoding Architecture*

The proposed video transcoding architecture that leverages the cloud services

A diagram of a company

Description automatically generated

*Preprocessor*

* 4 Responsibilities
  1. Video splitting – split video into smaller group of pictures (GOP) alignment.
     + GOP – group/chunk of frames arranged in a specific order
     + Each chunk is an independently playable unit, usually a few seconds in length
  2. Split videos by GOP alignment old clients
     + Some old mobile devices or browsers might not support video splitting
  3. DAG generation – generates DAG based on configuration files client programmers write
  4. Cache data – the preprocessor is a cache for segmented videos
     + For better reliability, the preprocessor stores GOPs, and metadata in temporary storage
     + If video encoding fails, the system could use persisted data for retry operations

*DAG Scheduler*

* Splits a DAG graph into stages of tasks and put them in the task queue in the resource manager
  + Original video is split into 3 stages – video, audio, and metadata
  + Video file is split into encoding and thumbnail

*Resource Manager*

* Responsible for managing the efficiency of resource allocation
* Contains 3 queues and a task scheduler
  + Task queue – contains tasks to be executed
  + Worker queue – contains worker utilization info
  + Running queue – contains info about current running tasks and workers running the tasks
  + Task scheduler – picks the optimal task/worker, and instructs the chosen task worker to execute the job

A diagram of a task scheduler

Description automatically generated

*Task Workers*

* Run tasked defined in the DAG
* Different task workers may run different tasks

*Temporary Storage*

* Multiple storage system types may be used
* Choice depends on data type, data size, access frequency, data life span
* Metadata
  + Frequently accessed by workers
  + Small data size
  + Cache in memory is a good option
* Video and audio data
  + Place in blob storage
* Data in temp storage is freed up once the video processing is complete

*Encoded Video*

* Final output of the encoding pipeline

*System Optimizations*

*Speed Optimizations*

* Parallelize video uploading
  + Split a video into smaller chunks by GOP alignment
  + Allows fast resumable uploads when the previous upload failed

A diagram of a company

Description automatically generated

* Add more geographically distributed centers
  + Set up multiple upload centers across the globe
  + US users uploads to the North America upload center
  + Users in China uploads to the Asian upload center
* Parallelism everywhere
  + Build loosely coupled systems
  + Introduce message queues
    - modules are no longer bottlenecked by the slowest steps
    - additional modules can be introduced at the slowest steps

*Safety Optimizations*

* Pre-signed upload URL
  + Ensure only authorized users upload videos to the right location
  + Client makes HTTP request to API servers to fetch pre-signed URL
  + Cloud services
    - Amazon S3
    - Microsoft Azure blob storage – named Shared Access Signature
* Content protection
  + Digital rights managements systems
    - Apple FairPlay
    - Google Widevine
    - Microsoft PlayReady
  + AES encryption – encrypt a video and configure an authorization policy
    - Decrypted upon playback
    - Ensures only authorized users can watch an encrypted video
  + Visual watermarking
    - Image overlap containing identity information (logo, company name)

*Cost Saving Optimizations*

* Only serve the most popular videos from CDN and other videos from our high-capacity storage video servers
* Short and unpopular videos can be encoded on-demand
* Some videos are popular only in certain regions and do not need international distribution
* Build your own CDN like Netflix and partner with Internet Service Providers (ISP)

*Error Handling*

*Types of Errors*

* Recoverable errors – when video segments fail to transcode, try again
  + Returns a proper error code when it continues to fail
* Non-recoverable errors – malformed video formats
  + System stops the running tasks associated with the video and return the proper error code to the client

*General solutions to errors*

|  |  |
| --- | --- |
| Upload error | Retry a few times |
| Split video error | If older version of clients cannot split videos by GOP alignment, the entire video is passed to the server |
| Transcoding error | Retry |
| Preprocessor error | Regenerate DAG diagram |
| Resource manager queue down | Use a replica |
| Task worker down | Retry the task on a new worker |
| API server down | Redirect requests to a different API server |
| Metadata cache server down | Data replication – if one node is down, you can still access other nodes to fetch data |
| Metadata DB server down | When master is down, promote one of the slaves |